Eingenfaces

1. Motivação

A abordagem eigenface começou com uma busca por uma representação de baixa dimensão de imagens de rosto. Sirovich e Kirby mostraram que a análise de componentes principais pode ser usada em uma coleção de imagens de rosto para formar um conjunto de características básicas. Essas imagens básicas, conhecidas como autoimagens, podem ser combinadas linearmente para reconstruir imagens no conjunto de treinamento original. Se o conjunto de treinamento consiste em M imagens, a análise de componentes principais pode formar um conjunto base de N imagens, onde $N < M$. O erro de reconstrução é reduzido aumentando o número de autoimagens; no entanto, o número necessário é sempre escolhido menor que M . Por exemplo, se você precisar gerar um número de Neigenfaces para um conjunto de treinamento de M imagens de rosto, você pode dizer que cada imagem de rosto pode ser composta de "proporções" de todos os K "features" ou eigenfaces

Em 1991, M. Turk e A. Pentland expandiram esses resultados e apresentaram o método de reconhecimento facial de eigenface. Além de projetar um sistema para reconhecimento de rosto automatizado usando autofaces, eles mostraram uma maneira de calcular os autovetores de uma matriz de covariância de modo que os computadores da época pudessem realizar a autodecomposição em um grande número de imagens de rosto. Imagens de rosto geralmente ocupam um espaço de alta dimensão e a análise convencional de componentes principais era intratável em tais conjuntos de dados. O artigo de Turk e Pentland demonstrou maneiras de extrair os autovetores com base em matrizes dimensionadas pelo número de imagens em vez do número de pixels.

Uma vez estabelecido, o método eigenface foi expandido para incluir métodos de pré-processamento para melhorar a precisão. Várias abordagens múltiplas também foram usadas para construir conjuntos de eigenfaces para diferentes assuntos e diferentes recursos, como os olhos.

2. Funcionamento

Este método consiste na extracção de toda a informação relevante da imagem facial analisada, codificação dessa informação o mais eficientemente possível e posterior comparação da face codificada com toda uma base de dados composta por faces codificadas de forma semelhante. Na realidade, é uma das formas mais intuitivas de classificar uma face.

Contrariamente a técnicas mais antigas, que se baseavam nas características particulares das faces, este método utiliza uma maior quantidade de informação, devido a classificar as faces com base em padrões faciais gerais. Esta técnica é semelhante à transformada de Fourier (FT), utilizada muito frequentemente em aplicações relacionadas com processamento de sinais, esta

operação matemática consiste na decomposição de uma função (sinal adquirido) em várias funções oscilatórias com parâmetros bem conhecidos (por exemplo, amplitude e frequência).

Este método de comparação facial consiste numa aproximação semelhante à FT. Cada face é decomposta numa série de componentes principais, ou vectores próprios da matriz covariância, definidos por um conjunto de faces de referência. Esta técnica foi desenvolvida por Sirovich e Kirby (1987) com o objectivo de representar, de forma eficiente, imagens de faces através da análise de componentes principais (PCA - Principal Component Analysis). Basicamente, cada face pode ser representada como a combinação linear de diversas Eigenfaces.

Assuma-se uma determinada imagem facial I(x,y) num espaço bidimensional N×N. Essa imagem pode ser encarada como um vector de dimensão N2. Nesse caso uma imagem com 256×256 pixéis passa a ser considerada um vector de dimensão 65,536, ou então um ponto num espaço com 65,536 dimensões. Como resultado, um conjunto de imagens pode mapear um grupo de pontos neste espaço gigantesco.

Todas as faces são semelhantes de uma forma geral (dois olhos, nariz, boca, ...). Como tal, não se vão distribuir de forma aleatória neste espaço enorme. Devido a essa similaridade, podem ser descritas por um subespaço relativamente pequeno. O objectivo deste método consiste em encontrar os vectores que melhor representam a distribuição das faces estudadas em todo o espaço da imagem. Esses vectores vão definir o subespaço das imagens faciais, designado por espaço facial.

Cada um desses vectores tem dimensão N2, descreve uma imagem N×N e consiste na combinação linear das imagens faciais originais. Uma vez que esses vectores são os vectores próprios (eigenvectors) da matriz covariância correspondente às imagens faciais originais, e uma vez que se assemelham a faces, são designados eigenfaces.

A Figura 2 inclui algumas imagens pertencentes a um conjunto de treino. Idealmente todas as imagens devem ter as mesmas dimensões (em pixéis) e estar representadas em gradiente cinza com valores entre 0 e 255. Cada imagem desse conjunto com M elementos representa um vector Γi (Γ1,Γ2,...,ΓM). A utilização de imagens diferentes para a mesma pessoa melhora a exactidão do processo (aumento de informação disponível para cada indivíduo).

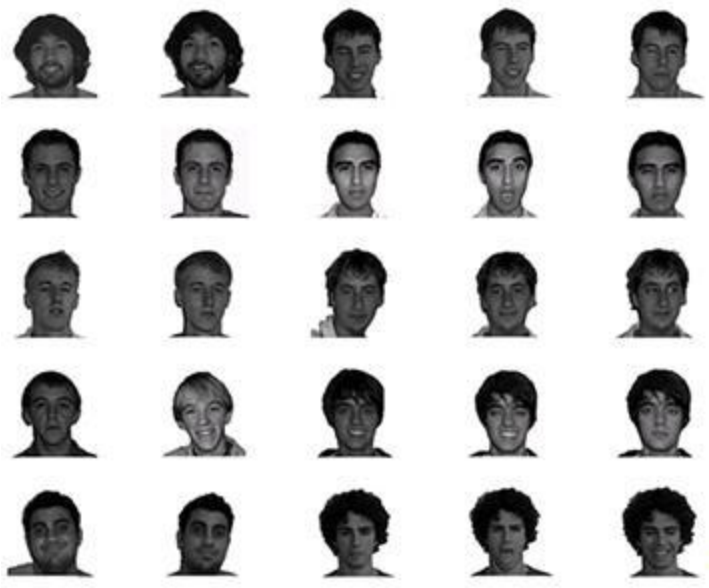


Figura 2. Exemplos de imagens faciais pertencentes a um conjunto de treino.

Em seguida, a face média (ψ) do conjunto é determinada através da expressão (1). Um exemplo do tipo de resultado obtido pode ser observado na Figura 3.

$$\psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n \qquad (1)$$

Figura 3. Face média (ψ) do conjunto de treino.

Cada uma das faces do conjunto de treino difere da média num vector ϕi=Γi-ψ. Este conjunto de vectores é posteriormente sujeito a uma análise de componentes principais, com o objectivo de determinar um conjunto com M vectores ortonormais (un) que melhor representem a distribuição dos dados. O vector de ordem k, uk, é escolhido de tal forma que (2) seja máximo:

$$\lambda_k = \frac{1}{M}\sum_{n=1}^{M}(u_k^T \phi_n)^2 \qquad (2)$$

tendo em conta que:

$$u_l^T u_k = \delta_{lk} = \begin{cases} 1, & \text{se } l = k \\ 0, & \text{caso contrário} \end{cases} \qquad (3)$$

Os vectores uk e os valores λk representam respectivamente os vectores e valores próprios da matriz covariância:

$$C = \frac{1}{M}\sum_{n=1}^{M}\phi_n \phi_n^T = AA^T \qquad (4)$$

Onde a matriz A=ϕ1 ϕ2 ... ϕM. O facto da matriz C ser de dimensão N2×N2, torna extremamente difícil determinar os seus N2 vectores próprios. Contudo, de acordo com a PCA, é possível reduzir o número de vectores próprios de N2 (dimensão do espaço das faces) para M (número de imagens do conjunto de treino). Uma vez que M≪N2, os cálculos ficam consideravelmente simplificados.

Torna-se então possível determinar a solução da matriz C, resolvendo primeiro uma matriz M×M e aplicando combinações lineares apropriadas das imagens faciais ϕi. Considerem-se os vectores próprios vi de AT A tais que:

$$A^T A v_i = \mu_i v_i \qquad (5)$$

Pré-multiplicando ambos os lados de (5) por A:

$$AA^T A v_i = \mu_i A v_i \qquad (6)$$

Ficando demonstrado que Avi são os vectores próprios de C=AAT. Com base nesta manipulação matemática é possível construir uma matriz L=AT A, onde

$$L_{mn} = \phi_m^T \phi_n \qquad (7)$$

e determinar os M vectores próprios (vi) de L. Estes vectores determinam as combinações lineares das M imagens do conjunto de treino que dão origem às eigenfaces (ui).

$$u_l = \sum_{k=1}^{N} v_{lk}\phi_k \qquad (8)$$

A Figura 4 representa as dez imagens (eigenfaces) com maior valor próprio associado. Ou seja, as que representam a maior variância no espaço das faces. Na prática é possível reconstruir qualquer imagem pertencente ao conjunto com apenas M' vectores próprios (M'≈0.25M).

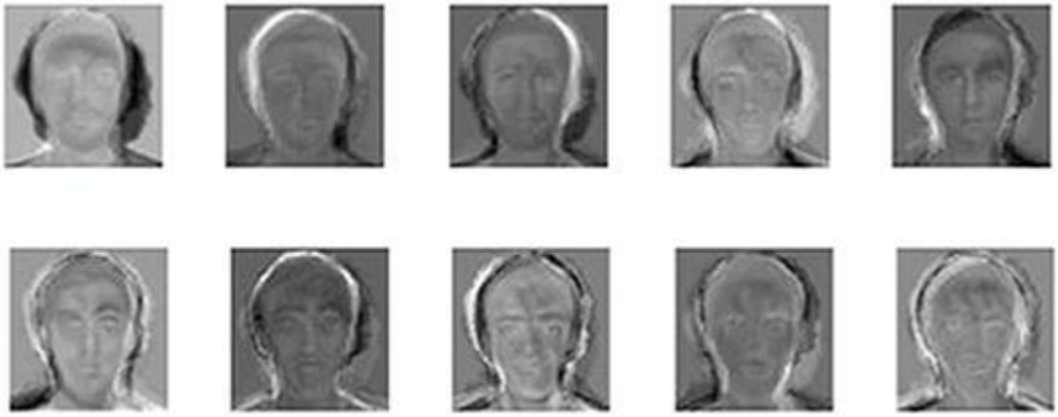


Figura 4. As dez eigenfaces com maior variância no espaço das faces definido pelo conjunto de treino, do qual as imagens apresentadas na Figura 2 fazem parte.

As eigenfaces determinadas a partir dos vectores próprios de L que apresentem maiores valores próprios podem então ser utilizadas para a classificação/identificação de faces. Para tal, a face a testar (Γ) é transformada nas suas componentes eigenface. Ou seja, é projectada no espaço das faces, através de:

$$\omega_k = u_k^T(\Gamma - \psi) \qquad (9)$$

Para k=1,2,...,M'. Os pesos (ωk) descrevem a contribuição de cada eigenface para a representação da imagem sob teste. O vector ΩT=ω1 ω2 ... ωM' pode então ser utilizado como algoritmo de reconhecimento de padrões para determinar qual a classe facial, caso exista alguma, que melhor descreve a face testada. Para tal, basta calcular a classe k que minimiza a distância euclidiana:

$$\epsilon_k^2 = \|(\Omega - \Omega_k)\|^2 \qquad (10)$$

Onde Ωk representa o vector que descreve a classe facial k. Cada classe facial Ωi é calculada através da média dos resultados da representação por eigenfaces de um pequeno número de faces (#faces=1 para maior precisão). Para além da distância a cada classe facial k (εk), é também possível calcular a distância entre a imagem testada e o espaço das faces:

$$\epsilon^2 = \|(\phi - \phi_f)\|^2 \qquad (11)$$

com:

$$\begin{cases} \phi = \Gamma - \psi \\ \phi_f = \sum_{i=1}^{M'} \omega_i u_i \end{cases} \qquad (12)$$

Existem então quatro resultados possíveis para qualquer imagem submetida:

Próxima do espaço facial e de alguma classe facial - indivíduo reconhecido e identificado;
Próxima do espaço facial mas não de uma classe facial - reconhecimento de um indivíduo desconhecido;
Distante do espaço facial e perto de uma classe facial - não é uma face (falso positivo);
Distante do espaço facial e de qualquer classe facial - não é uma face.
Um dos maiores problemas nos processos biométricos que envolvem reconhecimento facial é a pose facial. Basicamente, a posição da cabeça é um problema 3D para o reconhecimento facial de imagens 2D. Uma das propostas para solucionar este problema consiste em utilizar a técnica descrita anteriormente (eigenfaces) para determinar a posição da cabeça. Contrariamente à Figura 3, onde a posição das cabeças testadas deve ser mantida constante, a Figura 5 representa o tipo de conjunto de treino que deve ser utilizado.

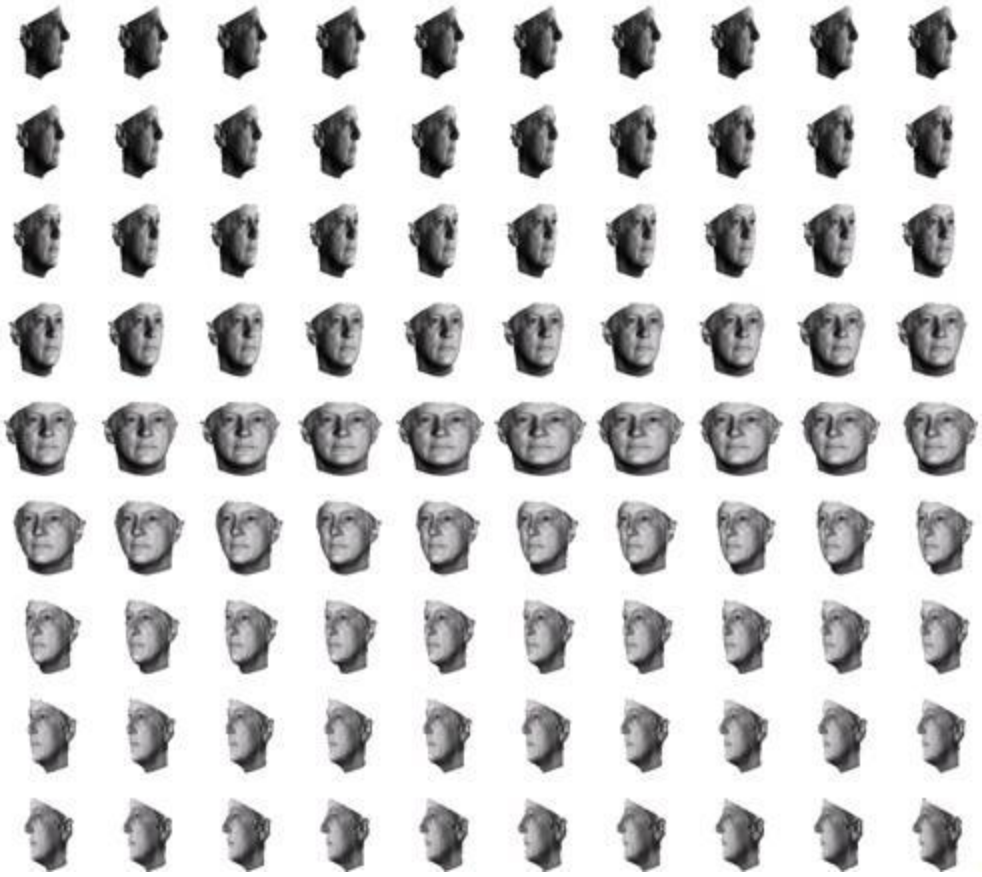


Figura 5. Amostra do conjunto de treino utilizado para reconhecimento da posição da cabeça.

Da mesma forma que no exemplo 2D, apenas uma parte dos elementos do conjunto de treino são necessários devido à variância das eigenfaces. Na Figura 6 encontram-se representadas as primeiras oito eigenfaces, das quais as primeiras três são suficientes para determinar uma boa estimativa da posição da cabeça.

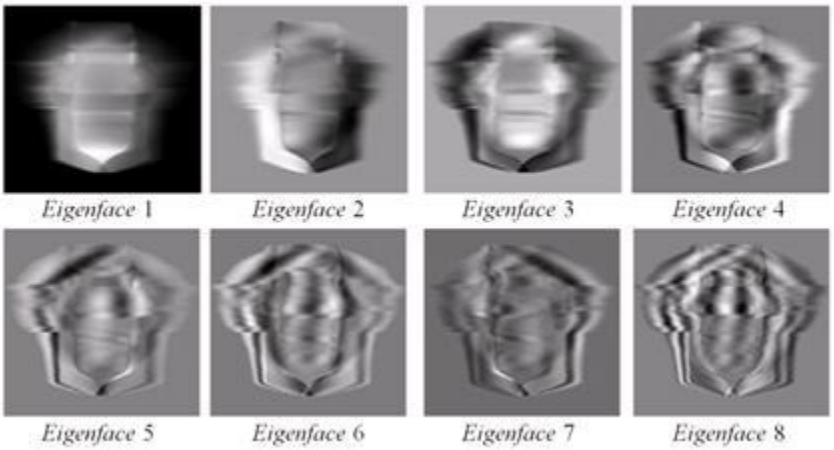


Figura 6. Eigenfaces com valor próprios mais elevados.

Resumindo, a abordagem por reconhecimento de eigenfaces pode ser dividida nos seguintes passos:

- Criação de uma biblioteca com imagens das faces de indivíduos característicos. Idealmente devem ser incluídas imagens diferentes para cada indivíduo, com variações na expressão facial e nas condições de iluminação.
- Cálculo da matriz L, determinação dos seus vectores e valores próprios, e determinação dos M' vectores próprios e respectivos valores próprios.
- Combinação do conjunto de imagens de treino normalizadas para determinação das M' eigenfaces uk.
- Cálculo e armazenamento do vector característico para cada indivíduo armazenado na biblioteca.
- Determinação da margem de erro (ϵk) que define a distância máxima permitida a qualquer classe facial. Opcionalmente, também pode ser escolhido um limite (ϵ) máximo para a distância ao espaço das faces.
- Para cada face identificada, o seu vector característico é calculado e comparado com os vectores característicos armazenados. Caso a comparação satisfaça as condições definidas no ponto anterior, para um membro no mínimo, a imagem é classificada como "conhecida". Caso contrário, a face analisada não é reconhecida. Logo, é classificada como "desconhecida", podendo optar-se pela sua adição à biblioteca (criação de uma nova classe facial).

Fisherfaces

1. Motivação

O Fisherfaces foi primeiramente utilizado em sistemas de reconhecimento de fala e posteriormente aplicado em sistemas de reconhecimento facial (FUJIKAWA, 2016), visando ser uma alternativa com maior acurácia ao Eigenfaces (CARNEIRO, 2012).

2. Funcionamento

•Esta abordagem maximiza o espalhamento entre classes e minimiza o espalhamento dentro das classes.
•Necessita que as amostras de treinamento estejam rotuladas com a classe a que pertençam.

•Cálculo da matriz de espalhamento entre classes

$$S_B = \sum_{i=1}^{c} N_i(\Psi_i - \Psi)(\Psi_i - \Psi)^T$$

Cálculo da matriz de espalhamento dentro das classes

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in X_i} (x_k - \Psi_i)(x_k - \Psi_i)^T$$

Cálculo da projeção W ótima•Maximiza a relação do determinante da matriz de espalhamento entre classes com o determinante da matriz dentro das classes obtendo-se os primeiro autovetorescom maiores autovalores.

$$W_{opt} = \arg\max_W \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|} = [\boldsymbol{w_1}, \boldsymbol{w_2}, \ldots, \boldsymbol{w_m}]$$

Devido ao problema da matriz Swser singular, usa-se a PCA para redução da dimensionalidade

$$W_{opt}^T = W_{fld}^T W_{pca}^T$$

onde:

$$W_{pca} = arg\max_W \left|W^T S_T W\right|$$

$$W_{fld} = arg\max_W \frac{\left|W^T W_{pca}^T S_B W_{pca} W\right|}{\left|W^T W_{pca}^T S_W W_{pca} W\right|} = arg\max_w \left(eig(S_W^{-1} S_b)\right)$$

Cálculo das projeções das faces no espaço de imagens

$$Y_i = W_{opt}^T (X_i - \Psi), \qquad i = 1, 2, \ldots, N.$$

Cada face do conjunto de treinamento (Yi) tem suas projeções calculadas e projetadas no espaço de imagens.•Uma face desconhecida tem suas projeções calculadas pela mesma equação acima e também é projetada no espaço de imagens.•A classe desconhecida é rotulada com a classe da imagem de treinamento a qual foi projetada mais próxima.•Para classificação pode ser utilizada a distância euclidiana para medir a distância entre as projeções e o classificador KNN, com K = 1.

3. Pontos positivos e Pontos negativos

A capacidade do algoritmo de Fisherfaces em identificar rostos e uma realidade, por ´ em´ quando estes estao em imagens onde as condic¸ ˜ oes n ˜ ao cooperam, esta se torna uma ta- ˜ refa complexa para o algoritmo. Sob a hipotese de que alterar estas imagens, de forma a ´ realc¸ar as informac¸oes mais importantes, pode melhorar o processo de reconhecimento, ˜ foram selecionadas soluc¸oes com ferramentas de c ˜ odigo aberto, renomadas na ´ area de ´ processamento de imagens. A aplicac¸ao destas objetivava auxiliar na identificac¸ ˜ ao de ˜ centenas de faces coletadas com o proposito de representar ambientes naturais, com fotos ´ de pessoas capturadas de forma nao planejada para serem reconhecidas por um algo- ˜ ritmo. Os metodos aplicados foram o filtro Bilateral, o filtro Gaussiano e a Equalizac¸ ´ ao˜ de Histograma, para analisar os resultados gerados por estas aplicac¸oes, realizando o re- ˜ conhecimento facial atraves do algoritmo de Fisherfaces. ´

Quando os filtros de abstrac¸ao foram aplicados de forma individual, imagens que ˜ nao eram reconhecidas puderam ser, apesar da quantidade de faces que se tornaram n ˜ ao˜ reconhecidas ter aumentado. Considerando o aumento nos resultados positivos associados a aplicac¸ao da Equalizac¸ ˜ ao de Histograma, pode-se perceber que a condic¸ ˜ ao da ˜ iluminac¸ao˜ e um fator crucial para a identificac¸ ´ ao, e que o m ˜ etodo demonstrou-se ´ util para ´

se obter mais chances de acerto. As combinac¸oes entre estes m ˜ etodos n ´ ao demonstraram ˜

impactos significativos: os resultados obtidos pelo filtro Gaussiano e Bilateral combinados intermediaram entre os resultados alcanc¸ados por estes individualmente, alcanc¸ando poucas identificac¸oes tamb ˜ em; todas as combinac¸ ´ oes com a Equalizac¸ ˜ ao de Histograma ˜

alcanc¸aram mais resultados positivos que o reconhecimento sem metodos, por ´ em nenhum ´ foi mais relevante que a normalizac¸ao do contraste exclusivamente, mesmo quando os tr ˜ esˆ foram combinados o efeito resultante foi similar.

Utilizar estes metodos para auxiliar o Fisherfaces pode ser recomendado para as- ´ pectos gerais, como a normalizac¸ao de contraste, ou as soluc¸ ˜ oes espec ˜ ´ıficas atendidas pelos filtros de abstrac¸ao, pois apesar dos impactos n ˜ ao serem t ˜ ao significantes, seu custo ˜

computacional, mesmo com uma grande quantidade de imagens, e´ ´ınfimo se comparado ao custo do treino do algoritmo de Fisherfaces, que aumenta junto com o tempo e proporcionalmente a quantidade de imagens utilizadas.

Local Binary Patterns Histograms (LBPH)

1. Motivação

Padrões binários locais (LBP) é um tipo de descritor visual usado para classificação em visão computacional . O LBP é o caso particular do modelo Texture Spectrum proposto em 1990. O

LBP foi descrito pela primeira vez em 1994. Desde então, descobriu-se que é um recurso poderoso para classificação de texturas; foi ainda determinado que quando LBP é combinado com o descritor Histograma de gradientes orientados (HOG), melhora consideravelmente o desempenho de detecção em alguns conjuntos de dados. Uma comparação de várias melhorias da LBP original no campo da subtração de fundo foi feita em 2015 por Silva et al. Uma pesquisa completa das diferentes versões da LBP pode ser encontrada em Bouwmans et al.

2. Funcionamento

Local Binary Pattern (LBP) é um operador de textura simples, porém eficiente, que rotula os pixels de uma imagem ao limitar a vizinhança de cada pixel e considera o resultado como um número binário.

Foi descrito pela primeira vez em 1994 (LBP) e, desde então, foi considerado um recurso poderoso para a classificação de textura. Ainda, quando o LBP é combinado com os histograms of oriented gradients (HOG), ele melhora o desempenho da detecção consideravelmente em alguns conjuntos de dados.

Usando o LBP combinado com histogramas, podemos representar as imagens do rosto como um vetor de dados simples.

Como o LBP é um descritor visual, ele também pode ser usado para tarefas de reconhecimento facial, como pode ser visto na seguinte explicação.

Agora que sabemos um pouco mais sobre o reconhecimento facial e o LBPH, vamos mais longe e vejamos as etapas do algoritmo:

1. Parâmetros: o LBPH usa 4 parâmetros:

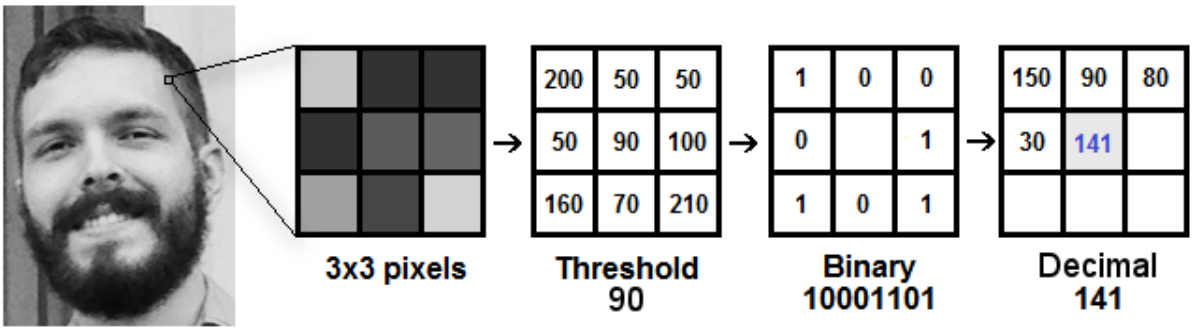
- Raio: o raio é usado para construir o padrão binário circular e representa o raio ao redor do pixel central. Geralmente, é definido como 1.
- Vizinhos: o número de pontos de amostra para construir o padrão binário circular local. Tenha em mente que: quanto mais pontos de amostra você incluir, maior será o custo computacional. Geralmente é definido como 8.
- Grade X: o número de células na direção horizontal. Quanto mais células mais fina é a grade e maior é a dimensionalidade do vetor de características resultante. Geralmente é definido como 8.
- Grade Y: o número de células na direção vertical. Quanto mais células, mais fina é a grade e maior é a dimensionalidade do vetor de características resultante. Geralmente é definido como 8.

2. Treinando o Algoritmo: Primeiro, precisamos treinar o algoritmo. Para fazer isso, precisamos usar um conjunto de dados com as imagens faciais das pessoas que queremos reconhecer.

Nós também precisamos definir um ID (pode ser um número ou o nome da pessoa) para cada imagem, então o algoritmo usará essas informações para reconhecer uma imagem de entrada e dar-lhe uma saída. Imagens da mesma pessoa devem ter o mesmo ID. Com o conjunto de treinamento já construído, vejamos os passos computacionais do LBPH.

3. Aplicando a operação LBP: O primeiro passo computacional do LBPH é criar uma imagem intermediária que descreva melhor a imagem original, destacando as características faciais. Para fazer isso, o algoritmo usa um conceito de janela deslizante, com base nos parâmetros raio e vizinhos.

A imagem abaixo mostra esse procedimento:

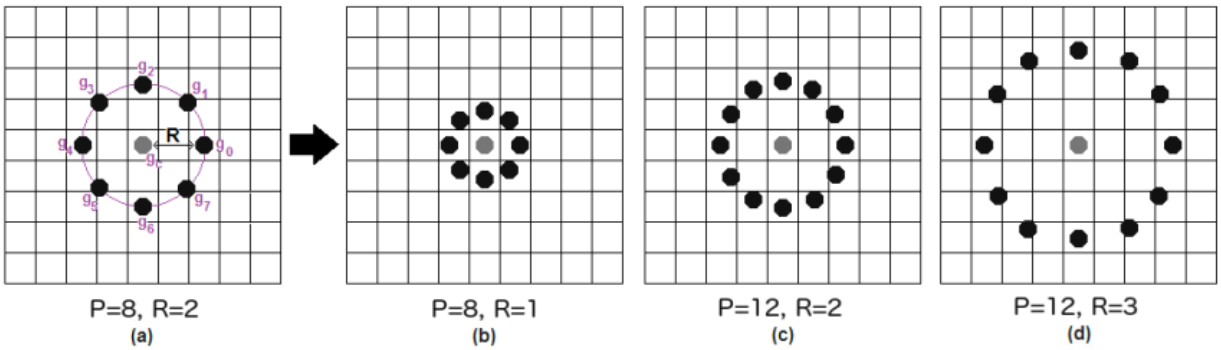


Com base na imagem acima, vamos dividir em várias pequenas etapas para que possamos entender isso facilmente:

- Suponha que tenhamos uma imagem facial em escala de cinza.
- Podemos obter parte desta imagem como uma janela de 3×3 pixels.
- Ele também pode ser representado como uma matriz 3×3 contendo a intensidade de cada pixel (0 ~ 255).
- Então, precisamos tomar o valor central da matriz para ser usado como limiar.
- Esse valor será usado para definir os novos valores dos 8 vizinhos.
- Para cada vizinho do valor central (limiar), estabelecemos um novo valor binário.
- Definimos 1 para valores iguais ou superiores aolimiar e 0 para valores inferiores aolimiar.
- Agora, a matriz conterá apenas valores binários (ignorando o valor central).
- Precisamos concatenar cada valor binário de cada posição da matriz linha por linha para um novo valor binário (por exemplo, 10001101). **Nota**: alguns autores usam outras abordagens para concatenar os valores binários (por exemplo, no sentido horário), mas o resultado final será o mesmo.

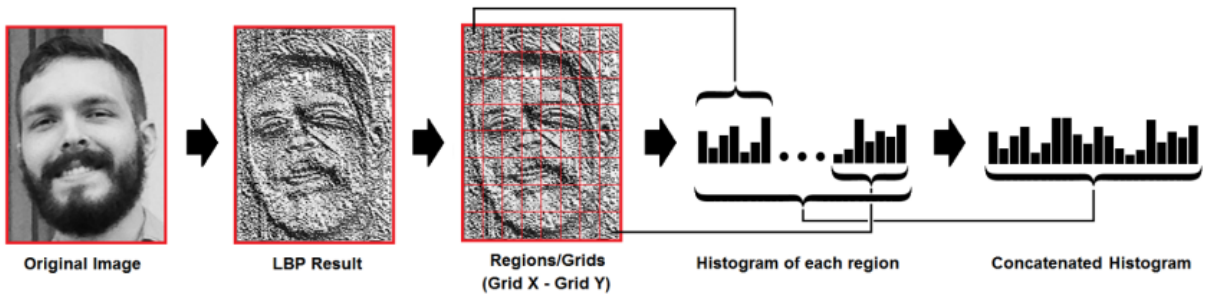
- Então, convertemos esse valor binário para um valor decimal e colocamos ele na posição central da matriz, que é realmente um pixel da nova imagem.
- No final deste procedimento (chamado LBP), temos uma nova imagem que representa melhor as características da imagem original.

Nota: O procedimento LBP foi expandido para usar um número diferente de raio e vizinhos, é chamado de Circular LBP.



Isso pode ser feito usando a interpolação bilinear. Se algum ponto de dados estiver entre os pixels, ele usa os valores dos 4 pixels mais próximos (2×2) para estimar o valor do novo ponto de dados.

4. Extraindo os histogramas: agora, usando a imagem gerada no último passo, podemos usar os parâmetros Grade X e Grade Y para dividir a imagem em múltiplas grades, como pode ser visto na imagem a seguir:



Com base na imagem acima, podemos extrair o histograma de cada região da seguinte maneira:

- Como temos uma imagem em escala de cinza, cada histograma (de cada grade) conterá apenas 256 posições (0 ~ 255) que representam as ocorrências de cada intensidade de pixel.
- Então, precisamos concatenar cada histograma para criar um histograma novo e maior. Supondo que tenhamos redes 8×8, teremos 8x8x256 = 16.384 posições no histograma final. O histograma final representa as características da imagem original da imagem.

O algoritmo LBPH é praticamente isso.

5. **Realizando o reconhecimento facial**: nesta etapa, o algoritmo já está treinado. Cada histograma criado é usado para representar cada imagem do conjunto de dados de treinamento. Assim, dada uma imagem de entrada, nós executamos as etapas novamente para esta nova imagem e criamos um histograma que representa a imagem.

Então, para encontrar a imagem que corresponde à imagem de entrada, precisamos comparar dois histogramas e devolver a imagem com o histograma mais próximo.

Podemos usar várias abordagens para comparar os histogramas (calcular a distância entre dois histogramas), por exemplo: distância euclidiana, qui-quadrado, valor absoluto, etc. Neste exemplo, podemos usar a distância euclidiana (que é bastante conhecida) baseada na seguinte fórmula:

Portanto, a saída do algoritmo é o ID da imagem com base no histograma mais próximo. O algoritmo também deve retornar a distância calculada, que pode ser usada como medida de “confiança”.
Nota: não se deixe enganar com o nome da “confiança”, pois as confianças mais baixas são melhores porque significa que a distância entre os dois histogramas é mais próxima.
Podemos usar um limite e a “confiança” para estimar automaticamente se o algoritmo reconheceu corretamente a imagem. Podemos assumir que a pessoa foi reconhecida com sucesso se a confiança for menor do que um limiar definido.

Conclusões

- LBPH é um dos algoritmos de reconhecimento facial mais fáceis de compreender.
- Pode representar características locais nas imagens.
- É possível obter excelentes resultados (principalmente em um ambiente controlado).
- É robusto contra transformações monotônicas em escala de cinza.
- É fornecido pela biblioteca OpenCV (Open Source Computer Vision Library)

Referencias:
https://updatedcode.wordpress.com/2017/11/26/reconhecimento-facial-como-funciona-o-lbph/#:~:text=Local%20Binary%20Pattern%20(LBP)%20%C3%A9,resultado%20como%20um%20n%C3%BAmero%20bin%C3%A1rio.

https://www.univates.br/bdu/bitstream/10737/2328/1/2018LuizHenriquedeOliveiraGalimberti.pdf

https://docplayer.com.br/22544884-Reconhecimento-facial-utilizando-fisherfaces-disciplina-reconhecimento-de-padroes-professor-andre-tavares-da-silva-mestrando-marcio-koch.html

http://www.sinfic.pt/SinficWeb/displayconteudo.do2?numero=44666

https://updatedcode.wordpress.com/2017/11/26/reconhecimento-facial-como-funciona-o-lbph/

https://pythonmachinelearning.pro/face-recognition-with-eigenfaces/

https://medium.com/@williangp/reconhecimento-de-padr%C3%B5es-eigenfaces-e4cef8f04919

http://www.acso.uneb.br/bahiart/uploads/BibFiles/ERBASE_WTICG_2015__Home_ReconhecimentoFacial.pdf